

Audit Report November, 2022

For

Vegas
ONE

Table of Content

Executive Summary	01
Checked Vulnerabilities	03
Techniques and Methods	04
Manual Testing	05
A. Contract - LPToken	05
High Severity Issues	05
Medium Severity Issues	05
Low Severity Issues	05
Informational Issues	05
Functional Testing	06
Automated Testing	08
Closing Summary	09
About QuillAudits	10

Executive Summary

Project Name VegasOne

Overview Stakingpool is masterchef inspired contract for staking NFTs. Whitelist checker contract manages access for NFTs and prices oracles across other blockchains. The purpose of stakingPool contracts is to incentivize stakers of NFTs in ERC20 tokens who holds the whitelisted NFTs. Users can withdraw rewards after staking or all at once if they're done with it.

Timeline 19 september, 2022 to 3rd october, 2022

Method Manual Review, Functional Testing, Automated Testing etc.

Scope of Audit The scope of this audit was to analyse VegasOne Staking Pool codebase for quality, security, and correctness.

<https://github.com/taisys-technologies/audit-stakingpool>

Commit hash: 79ddedfa16d1c8ca0c02df89db70cbe6d148643c

0
Issues Found

High

Medium

Low

Informational

	High	Medium	Low	Informational
Open Issues	0	0	0	0
Acknowledged Issues	0	0	0	0
Partially Resolved Issues	0	0	0	0
Resolved Issues	0	0	0	0



Types of Severities

High

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality, and we recommend these issues be fixed before moving to a live environment.

Medium

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems, and they should still be fixed.

Low

Low-level severity issues can cause minor impact and or are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

Informational

These are severity issues that indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

Types of Issues

Open

Security vulnerabilities identified that must be resolved and are currently unresolved.

Resolved

These are the issues identified in the initial audit and have been successfully fixed.

Acknowledged

Vulnerabilities which have been acknowledged but are yet to be resolved.

Partially Resolved

Considerable efforts have been invested to reduce the risk/impact of the security issue, but are not completely resolved.



Checked Vulnerabilities

- ✓ Re-entrancy
- ✓ Timestamp Dependence
- ✓ Gas Limit and Loops
- ✓ Exception Disorder
- ✓ Gasless Send
- ✓ Use of tx.origin
- ✓ Compiler version not fixed
- ✓ Address hardcoded
- ✓ Divide before multiply
- ✓ Integer overflow/underflow
- ✓ Dangerous strict equalities
- ✓ Tautology or contradiction
- ✓ Return values of low-level calls
- ✓ Missing Zero Address Validation
- ✓ Private modifier
- ✓ Revert/require functions
- ✓ Using block.timestamp
- ✓ Multiple Sends
- ✓ Using SHA3
- ✓ Using suicide
- ✓ Using throw
- ✓ Using inline assembly



Techniques and Methods

Throughout the audit of smart contract, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments match logic and expected behaviour.
- Token distribution and calculations are as per the intended behaviour mentioned in the whitepaper.
- Implementation of ERC-20 token standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods and tools were used to review all the smart contracts.

Structural Analysis

In this step, we have analysed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.

Static Analysis

Static analysis of smart contracts was done to identify contract vulnerabilities. In this step, a series of automated tools are used to test the security of smart contracts.

Code Review / Manual Analysis

Manual analysis or review of code was done to identify new vulnerabilities or verify the vulnerabilities found during the static analysis. Contracts were completely manually analysed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of the automated analysis were manually verified.

Gas Consumption

In this step, we have checked the behaviour of smart contracts in production. Checks were done to know how much gas gets consumed and the possibilities of optimization of code to reduce gas consumption.

Tools and Platforms used for Audit

Remix IDE, Truffle, Truffle Team, Solhint, Mythril, Slither, Solidity statistic analysis.



Manual Testing

A. Contract - StakingPools.sol, Whitelist.sol and WhitelistChecker.sol

High Severity Issues

No issues found

Medium Severity Issues

No issues found

Low Severity Issues

No issues found

Informational Issues

No issues found



Functional Testing

interfaces		
<i>IDetailedERC20.sol</i>	interface	PASS
<i>IMintableERC20.sol</i>	interface	PASS
<i>IWhitelistChecker.sol</i>	interface	PASS
libraries		
<i>Pool.sol</i>		
Period	struct	PASS
Level	struct	PASS
Context	struct	PASS
Data	struct	PASS
toPeriods	function	PASS
updatePeriod	function	PASS
<i>Stake.sol</i>		
Data	struct	PASS
canClaim	function	PASS
_updateInterest	function	PASS
update	function	PASS
contracts		
<i>WhitelistChecker.sol</i>		
onlyGovernance	modifier	PASS
setPendingGovernance	function	PASS
acceptGovernance	function	PASS
inWhitelist	modifier	PASS
addWhitelist	function	PASS



removeWhitelist	function	PASS
getAllTokens	function	PASS
getAllBalance	function	PASS
<i>Whitelist.sol</i>		
constructor	constructor	PASS
<i>onlyGovernance</i>	modifier	PASS
setPendingGovernance	function	PASS
acceptGovernance	function	PASS
addChecker	function	PASS
removeChecker	function	PASS
addNFT	function	PASS
removeNFT	function	PASS
includeOwner	function	PASS
balanceOf	function	PASS
listTokens	function	PASS
<i>_StakingPools.sol</i>		
constructor	constructor	PASS
acceptsRemoval	function	PASS
updatePeriod	function	PASS
<i>setPeriodThreshold</i>	function	PASS
setRewardingAddress	function	PASS
addLevel	function	PASS
deposit	function	PASS
claim	function	PASS
exit	function	PASS

getPeriodCount	function	PASS
getPeriodThreshold	function	PASS
getRewardingAddress	function	PASS
getLevelCount	function	PASS
<i>getLevel</i>	function	PASS
canClaim	function	PASS
<i>getPoolToken</i>	function	PASS
getPoolTotalDeposited	function	PASS
getStakeTotalDeposited	function	PASS
getStakeInfo	function	PASS
_deposit	function	PASS
_withdraw	function	PASS
_claim	function	PASS
_inLevel	function	PASS

Automated Tests

No major issues were found. Some false positive errors were reported by the tools. All the other issues have been categorized above according to their level of severity.

Closing Summary

In this report, we have considered the security of the VegasOne Staking Pool Contract. We performed our audit according to the procedure described above.

No Issues Found During the Audit.

Disclaimer

QuillAudits smart contract audit is not a security warranty, investment advice, or an endorsement of the VegasOne Platform. This audit does not provide a security or correctness guarantee of the audited smart contracts.

The statements made in this document should not be interpreted as investment or legal advice, nor should its authors be held accountable for decisions made based on them. Securing smart contracts is a multistep process. One audit cannot be considered enough. We recommend that the VegasOne Team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.



About QuillAudits

QuillAudits is a secure smart contracts audit platform designed by QuillHash Technologies.

We are a team of dedicated blockchain security experts and smart contract auditors determined to ensure that Smart Contract-based Web3 projects can avail the latest and best security solutions to operate in a trustworthy and risk-free ecosystem.



700+

Audits Completed



\$15B

Secured



700K

Lines of Code Audited



Follow Our Journey



Audit Report November, 2022

For

Vegas
ONE



QuillAudits

📍 Canada, India, Singapore, United Kingdom

🌐 audits.quillhash.com

✉️ audits@quillhash.com